Laporan Membangun Pos Tagger
Bahasa Indonesia

Ditunjukan untuk memenuhi tugas 2 Pemrosesan Bahasa Alami terkait *Language Modeling*.

Oleh :

Alfian Yulianto – 1301178160

Akbar Habib Buana Wibawa Putra – 1301178198

Bastomy – 1301178418

Muhammad Hanafiah – 1301178552

ICM-39-GAB

S1 Teknik Informatika

Fakultas Informatika

Telkom University

2018

Untuk membangun sebuah POSTagger, terdapat beberapa pendekatan yang dapat diaplikasikan. Dalam tugas 2 mata kuliah pemrograman bahasa alami ini ada 3 pendekatan yang digunakan, yaitu:

1. Baseline
   Baseline adalah salah satu pendekatan yang digunakan untuk membangun suatu postagger. Pendekatan Baseline ini menggunakan probabilitas frekuensi kemunculan tag pada suatu kata. Jika terdapat lebih dari satu tag yang memiliki frekuensi yang sama pada satu kata, maka akan di pilih sembarang tag. Kata yang tidak terdapat pada korpus latih akan diberikan tag “UNK” atau unknown.
2. Decision Tree
   Pendekatan ini di bangun menggunakan library dari sklearn. Pada metode ini untuk melakukan ekstraksi fitur perlu dilakukan pembersihan terhadap korpus latih yang sebelumnya sudah terdapat tag.
3. HMM - Viterbi
   adalah sebuah kakas statistik yang sudah banyak diterapkan di bidang signal processing, dan speech processing. Pada pendekatan ini dilakukan perhitungan frekuensi setiap postag, frekuensi transisi tag, dan frekuensi emisi. Setelah itu dihitung probabilitas transisi dengan persamaan: P(t1|t2) = C(t1|t2) / C(t1), frekuensi bigram transisi per frekuensi tag pertama, dan probabilitas emisi dengan persamaan P(t|w) = C(t|w)/C(t), frekuensi bigram tag dan kata per frekuensi tag. Pencarian POS tag pada HMM - Viterbi dilakukan dengan cara menghitung probabilitas path.

Analisa

Pada program ini menggunakan 1000 kalimat training dan 20 kalimat testing.

Akurasi yang didapat dengan menggunakan pendekatan Baseline

```
print("akurasi ", akurasi," persen")
```

```
akurasi  87.97595190380761  persen
```

Akurasi yang di dapat dengan menggunakan pendekatan decision tree

```
X_test, y_test = transform_to_dataset(testing_sentences, testing_tags)

print("Accuracy:")
print(clf.score(X_test, y_test)*100,'persen')
```

```
Accuracy:
94.7895791583 persen
```

Kesimpulan: dari beberapa pendekatan yang digunakan, akurasi tertinggi adalah Decision Tree karna pada pendekatan ini dicari peluang kemunculan suatu tag setelah tag yang lain dan juga

peluang suatu kata memiliki tag tersebut sehingga lebih akurat dikarnakan pendekatan ini dapat melihat satu kata berikutnya (bigram) tidak seperti baseline yang hanya melihat peluang satu kata memiliki tag tertentu (unigram).